Amendments to the Claims:

This listing of claims will replace all prior versions, and listing of claims in the application:

Listing of the Claims:

1. (Previously Presented) A method of generating a Java macro instruction corresponding to one or more Java Bytecode instructions, said method comprising:

reading a stream of Java Bytecode instructions;

determining whether two or more Java Bytecode instructions in said Java Bytecode stream can be represented by one instruction;

generating a Java macro instruction that represents said two or more Java Bytecode instructions when said determining determines that two or more Java Bytecode instructions in said Java Bytecode stream can be represented by one instruction, wherein said Java macro instruction is suitable for execution by a Java virtual machine;

generating an internal representation of said Java macro instruction in a pair of streams that collectively represent an internal representation of said stream of Java Bytecode instructions in said Java virtual machine; and

wherein when executed said Java macro instruction can operate to perform one or more operations that are performed by said two or more Java Bytecode instructions.

2. (Original) A method as recited in claim 1,

wherein said determining operates to determine whether a predetermined sequence of two or more Java Bytecode instructions have been found.

3. (Previously Presented) A method as recited in claim 1, wherein said determining is performed during Java Bytecode verification by said virtual machine.

4-5. (Cancelled)

6. **(Currently Amended)** A method as recited in claim [[5]] **1**,

wherein said pair of streams includes a code stream and a data stream,

wherein said code stream is suitable for containing a code portion of said Java macro instruction, and

wherein said data stream is suitable for containing a data portion of said Java macro instruction.

7. (Previously Presented) A method of generating a Java macro instruction corresponding to one or more Java Bytecode instructions, said method comprising:

reading a stream of Java Bytecode instructions;

counting, during Bytecode verification, the number of times a sequence of Java Bytecode instructions appears in said stream of Java Bytecode instructions, said sequence of Java Bytecode instructions including two or more Java Bytecode instructions which are in a sequence in said stream;

determining, during Bytecode verification, whether said sequence of Java Bytecode instructions should be represented by one instruction;

generating, during Bytecode verification, a Java macro instruction that represents said sequence of Java Bytecode instructions when said determining determines that said sequence of Java Bytecode instructions can be represented by said one instruction;

wherein said Java macro instruction is suitable for execution by a Java virtual machine; and

wherein when executed said Java macro instruction can operate to perform one or more operations that are performed by said sequence of Java Bytecode instructions.

8. (Original) A method as recited in claim 7, wherein said determining of whether said sequence of Java Bytecode instructions should be represented by one instruction operates to determine whether said sequence has been counted for at least a predetermined number of times.

9. (Canceled)

10. (Currently Amended) A method as recited in claim [[9]] **1**, wherein said method further comprises:

replacing said two or more Java Bytecode instructions with said Java macro instruction, and

wherein said macro instruction is internally represented in said virtual machine.

11. (Original) A method as recited in claim 10, wherein said internal representation comprises a pair of streams.

12. (Original) A method as recited in claim 11,

wherein said pair of streams includes a code stream and a data stream,

wherein said code stream is suitable for containing a code portion of said Java macro instruction, and

wherein said data stream is suitable for containing a data portion of said Java macro instruction.

13. (Currently Amended) A method of generating a Java macro instruction corresponding to one or more Java Bytecode instructions, said method comprising:

reading a stream of Java Bytecode instructions during Java Bytecode verification;

determining, **during** Java Bytecode verification, whether two or more Java Bytecode instructions in said Java Bytecode stream can be represented by one instruction;

generating a Java macro instruction, **during** Java Bytecode verification, that represents said two or more Java Bytecode instructions when said determining determines that two or more Java Bytecode instructions in said Java Bytecode stream can be represented by one instruction;

wherein said Java macro instruction is suitable for execution by a Java virtual machine; and

wherein, when executed, said Java macro instruction can operate to perform one or more operations that are performed by said two or more Java Bytecode instructions.

14. (Original) A method as recited in claim 13,

wherein said determining operates to determine whether a predetermined sequence of two or more Java Bytecode instructions have been found.

15. (Original) A method as recited in claim 13,

wherein said method further comprises counting the number of times a sequence of Java Bytecode instructions appear in said stream, and

wherein said determining operates to determine whether a sequence has been counted for at least a predetermined number of times.

16. (Previously Presented) In a Java computing environment, a Java macro instruction generator suitable for generation of Java macro instructions,

wherein each Java macro instruction corresponds to one or more Java Bytecode instructions,

wherein said Java macro instruction generator operates to:

read a stream of Java Bytecode instructions during Java Bytecode verification;

determine whether two or more Java Bytecode instructions in said Java Bytecode stream can be represented by one instruction;

generate a Java macro instruction that represents said two or more Java Bytecode instructions when said determining determines that two or more Java Bytecode instructions in said Java Bytecode stream can be represented by one instruction, wherein said Java macro instruction is suitable for execution by a Java virtual machine,

generate an internal representation of said Java macro instruction in a pair of streams that collectively represent an internal representation of said stream of Java Bytecode instructions in said Java virtual machine; and

wherein, when executed, said Java macro instruction can operate to perform one or more operations that are performed by said two or more Java Bytecode instructions.

17. (Original) A Java macro instruction generator as recited in claim 16, wherein said Java macro instruction generator operates during Java Bytecode verification.

18. (Original) A Java macro instruction generator as recited in claim 16, wherein said Java macro instruction generator operates to determine whether a predetermined sequence of two or more Java Bytecode instructions has been found.

19-20. (Canceled)